

Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended): A method comprising:

generating a functional design of a logic circuit by selecting, placing, and connecting ~~reusable graphical library~~ elements using a graphical user interface, the graphical ~~library~~ elements representing ~~logical functions~~ elements of the logic circuit, [[and]] the elements comprising connections between the logical functions state elements, combinatorial logic, reusable library elements, and elements containing textual description of logic;

refining the functional design to represent a hardware design of the logic circuit using the graphical user interface to modify the connectivity of the graphical elements;

maintaining a data structure representative of ~~a model~~ the functional design, the model including combinational blocks, state elements and graphical library elements of the logic circuit; and

generating an architectural, cycle-based simulation model of the functional design of the logic circuit and ~~an~~ separate implementation hardware description language (HDL) model of the hardware design of the logic circuit from the data structure, the implementation HDL model functionally equivalent to the architectural, cycle-based simulation model.

2. (Original): The method of claim 1 wherein the data structure comprises a description of a net list.

3. (Original): The method of claim 2 wherein the data structure comprises:

elements representing logical functions;

elements representing connection points to gates;

elements representing all bits of a simulation state; and

elements representing an arbitrary collection of bits within the simulation state.

4. (Currently Amended): The method of claim 3 wherein the elements comprise ~~are all~~ C++ classes used to generate the architectural, cycle-based simulation model.

5. (Currently Amended): The method of claim 1 wherein the architectural, cycle-based simulation model comprises C++ software code.

6. (Cancelled)

7. (Previously presented): The method of claim 1 wherein the HDL is Verilog.

8. (Previously presented): The method of claim 1 wherein the HDL is Very high speed integrated circuit Hardware Design Language (VHDL).

9. (Currently Amended): A method comprising:

generating a model containing combinatorial blocks, state elements, ~~[[and]]~~ reusable ~~graphical~~ library elements, and elements containing textual description of logic using a graphical user interface, the model representing a functional design of a logic circuit;

refining the model to represent a hardware design of the logic circuit using the graphical user interface;

maintaining a descriptive net list of the model; and

generating a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

10. (Cancelled)

11. (Original): The method of claim 9 wherein the net list comprises gates, nodes and nets.

12. (Currently Amended): The method of claim 9 wherein maintaining comprises parsing and analyzing the combinatorial blocks, state elements, ~~and graphical~~ reusable library elements, ~~and elements containing textual description of logic~~ of the model.

13. (Currently Amended): The method of claim 9 wherein generating ~~[[a]]~~ the C++ architectural, cycle-based simulation model and ~~[[a]]~~ the functionally equivalent implementation Verilog model comprises:

partitioning a topology of the net list into a plurality of partitions; and
code ordering each of the partitions.

14. (Currently Amended): A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

generate a model containing combinatorial blocks, state elements, ~~[[and]]~~ reusable ~~graphical~~ library elements, and elements containing textual description of logic using a graphical user interface, the model representing a functional design of the logic circuit;
refine the model to represent a hardware design of the logic circuit;
maintain a descriptive net list of the model; and
generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

15. (Original): The computer product of claim 14 wherein the computer readable medium is a random access memory (RAM).

16. (Original): The computer product of claim 14 wherein the computer readable medium is a read only memory (ROM).

17. (Original): The computer product of claim 14 wherein the computer readable medium is a hard disk drive.

18. (Currently Amended): A processor and memory configured to:

generate a model containing combinatorial blocks, state elements, ~~[[and]]~~ reusable ~~graphical~~ library elements, and elements containing textual description of logic using a graphical user interface, the model representing a functional design of the logic circuit;
refine the model to represent a hardware design of the logic circuit;
maintain a descriptive net list of the model; and
generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

19. (Original): The processor and memory of claim 18 wherein the processor and memory are incorporated into a personal computer.

20. (Original): The processor and memory of claim 18 wherein the processor and memory are incorporated into a network server residing in the Internet.

21. (Original): The processor and memory of claim 18 wherein the processor and memory are incorporated into a single board computer.

22. (Currently Amended): A system comprising:

a graphic user interface (GUI) for receiving selections of ~~reusable~~ graphical library elements to generate a model and displaying the model, the model containing combinatorial blocks, state elements, ~~and graphical~~ library elements, and elements containing textual description of logic, the model representing a functional, performance based model and a functionally equivalent implementation hardware design of a logic circuit;

a maintenance process to manage a data structure representing a descriptive net list of the model; and

a code generation process to generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the data structure.

23. (Original): The system of claim 22 wherein the data structure comprises gates, nodes and nets.

24. (Currently Amended): The system of claim 22 wherein the maintenance process comprises parsing and analyzing the combinatorial blocks, state elements, ~~and graphical~~ reusable library elements, and elements containing textual description of logic of the model.

25. (Original): The system of claim 22 wherein the code generation process comprises:
partitioning a topology of the net list into a plurality of partitions; and
code ordering each of the partitions.

26. (Withdrawn): A data structure comprising:
elements representing logical functions of a logic model;
elements representing connection points to gates of the logic model;
elements representing all bits of a simulation state of the logic model; and
elements representing an arbitrary collection of bits within the simulation state of the logic model.

27. (Withdrawn): The data structure of claim 26 wherein the elements are stored in a binary tree.

28. (Withdrawn): The data structure of claim 26 wherein the elements are stored in a linked list.